Amendments to the Claims

This listing of claims will replace all prior versions and listings of claims in the application:

Listing of Claims:

What is Claimed is:

1. (Canceled)
2. (Canceled)
3. (Canceled)
4. (Canceled)
5. (Currently Amended) A hardware engine ~~system~~ for completely offloading Transmission Control Protocol/Internet Protocol (TCP/IP) protocol processing both in a transmit path and a receive path from a host system software stack, ~~transmitting and receiving TCP/IP data packets using hardware engine~~, comprising:

an inbound MAC Receive state machine for receiving ~~processing MAC frames received from~~ a network packet and passing the network packet to an inbound memory buffer;

an inbound IP verifier hardware state machine for verifying IP packet headers if the received packet from the network is an IP packet;

an inbound IP fragment processing state machine for processing and reassembling IP packet fragments into an IP datagram that is stored in a local memory for the hardware engine; wherein the inbound IP fragment processing state machine assigns a timer to each IP datagram with a timeout value, and if a timeout occurs then the entire datagram is removed from a reassembly list and storage space associated with the timed out datagram is assigned to a free buffer list that is maintained by a buffer list manager, where the buffer list manager implements a hardware state machine to manage storage space in the local memory and

grants free buffer space when available to the inbound IP verifier hardware state machine; and

an inbound TCP hardware state machine for (a) processing TCP segments received from an IP layer by completely offloading TCP/IP protocol stack processing from the host system software stack to hardware; (b) re-ordering out of order TCP segments and then transferring the TCP data to the host system; and (c) transferring data from an iSCSI connection to a processor that processes iSCSI frames; wherein the inbound TCP hardware state machine retrieves a network control block from a TCP table manager and updates network state information in the network control block and maintains a segment reassembly list for each network connection;

wherein the inbound IP verifier state machine passes non-IP data packets to [[a]] the host system via an inbound direct memory access engine (IDE).

6. (Currently Amended) The hardware engine ~~system~~ of Claim 5, wherein the inbound IP verifier hardware state machine verifies IP packet header information and if the header information is valid, then temporarily stores the packet in ~~an external~~ the local memory and if the header information is not valid then the packet is discarded and an address for a memory buffer for the discarded packet is passed to the buffer list manager.

7. (Currently Amended) The hardware engine ~~system~~ of Claim 5, wherein the inbound IP verifier state machine passes complete IP datagrams to the host that are non-TCP packets via the IDE .

8. (Canceled)

9. (Canceled)

10. (Canceled)

11. (Currently Amended) A system for processing network data packets using a hardware engine coupled to a host system and a network, comprising:

a verification module that verifies incoming network data packets;

a first in-bound transmission control protocol (TCP) processor for (a) processing TCP segments received from a network by completely offloading a TCP/IP (Internet Protocol) protocol stack processing from a host system software stack to hardware; (b) re-ordering out of order TCP segments and then transferring TCP data to the host system; and (c) transferring data from an iSCSI connection to a second in-bound processor that processes iSCSI frames and non-TCP data; wherein the first in-bound TCP processor retrieves network control blocks and updates network state information in the network control block and maintains a segment reassembly list for each network connection;

a first outbound processor that obtains connection state information from a network control block; builds a TCP header and sends TCP data to the network;

a second outbound processor that processes media access control (MAC) and IP transfer requests;

a fragment processor that receives data packet fragments and reassembles the data packet fragments into complete datagrams for delivery; wherein the fragment processor assigns a timer to each datagram with a timeout value and if a timeout occurs then the entire datagram is removed from a reassembly list and storage space associated with the timed out datagram is assigned to a free buffer list that is maintained by a buffer list manager, where the buffer list manager implements a hardware state machine to manage storage space in a local memory and grants free buffer space when available to the verification module; [[and]]

a third outbound processor for processing small computer system interface (SCSI) requests received from the host system via a hardware TCP stack; and

a TCP Table manager that interfaces with the first outbound TCP processor, the first in-bound TCP processor, the second in-bound processor and the third outbound processor; the TCP Table Manager maintains timer functions for all TCP connections at any given time in the hardware engine and maintains a linked list of network control blocks for processing network data sent by the hardware engine and received by the hardware engine.

~~a second in-bound processor for processing incoming TCP segments destined for iSCSI.~~

12. (Canceled)

13. (Currently Amended) The system of Claim 11[[2]], wherein the second outbound processor builds a header for IP data requests based on entries from a network control block maintained at the local memory ~~also acts as a pass through processor for TCP data processed by the first outbound processor~~.

14. (Currently Amended) The system of Claim 11[[2]], wherein the first outbound TCP processor builds TCP header data and passes the header data to the second outbound processor.

15. (Currently Amended) The system of Claim 11, wherein the verification module passes non-IP data packets to a host via an inbound direct memory access module.

16. (Currently Amended) The system of Claim 11, wherein the verification module verifies IP data packet header information and if the header information is valid, then the data packet is added to a list maintained by the verification module at the local memory.

17. (Canceled)

18. (Currently Amended) The system of Claim 11, wherein the fragment processor sets a flag if overlapping datagrams are received, and the flag indicates when a TCP checksum is to be performed ~~must be re-run~~.

19. (Currently Amended) The system of Claim 11, wherein the first inbound TCP processor re-orders out of order data segments.

20. (Currently Amended) The system of Claim 11, wherein the first inbound TCP processor maintains a segment re-assembly list for each network connection and is linked with a network control block.

21. (Currently Amended) The system of Claim 11, wherein the first inbound TCP processor includes a receive block that receives data; a validation block that validates data segments; and an option block for validating a TCP timestamp that is found in TCP option data; and an acknowledgement processor that performs TCP acknowledgement processing.

22. (Currently Amended) The system of Claim 11, wherein the first inbound TCP processor receives a data segment with a TCP header and option data.

23. (Canceled)

24. (Currently Amended) A system for processing incoming TCP data packets received by a hardware engine that is coupled to a network and a host system, comprising:

an input TCP processing module that includes:

(a) an input processing module that determines if a TCP connection is established and checks for TCP flags to determine if a TCP data packet should be processed; wherein the input processing module obtains a network control block from a TCP table manager for a valid TCP data packet; and the TCP table manager maintains timer functions for all TCP connections at any given time in the hardware engine and maintains a linked list of network control blocks for processing network data sent by the hardware engine and received by the hardware engine ;

(b) an acknowledgement processor module that interfaces with the input processing module, where the acknowledgement processor module (i) handles any acknowledgement information included in the TCP data packet; and (ii) updates TCP congestion window for a TCP connection; and

(c) a Data processor module that handles any data included in the TCP data packet and if a TCP data packet does not have to be discarded or sent to the host system; wherein the first input processing module validates and saves TCP timestamps by checking if a received timestamp is greater than a most recently saved timestamp ; and wherein the hardware engine completely offloads TCP/IP protocol processing from a host system software.

25. (Currently Amended) The system of Claim 24, wherein the Data Processor module determines if a received packet was in order or out of order and trims the packet if it requires trimming before sending the packet to a destination.

26. (Currently Amended) The system of Claim 24, wherein the TCP Table Manager provides a read / write register interface to allow simultaneous access to fields within the network control blocks ~~TCP connection state is organized in network control blocks and stored in a local memory~~.

27. (Canceled)

28. (Canceled)

29. (New) A hardware engine for offloading transmission control protocol/Internet Protocol processing from a software stack at a host system, comprising:

a TCP Table manager that interfaces with (a) an outbound TCP processor for processing TCP packets sent by the host system, (b) an inbound TCP processor for processing TCP packets received from a network device, (c) an inbound processor for processing non-TCP network packets received from a network device and (d) an outbound processor for processing non-TCP packets sent by the host system, where the TCP Table Manager maintains timer functions for all TCP connections at any given time in the hardware engine and maintains a linked list of network control blocks for processing network data sent by the hardware engine and received by the hardware engine, wherein the hardware engine completely offloads execution of a TCP/IP protocol stack in hardware from a host system that executes the TCP/IP protocol stack in software.

30. (New) The hardware engine of Claim 29, wherein the outbound TCP processor requests the TCP Table Manager to read a network control block for an existing TCP connection stored at local memory of the hardware engine and place the network control block in a register set maintained by the TCP Table Manager and accessible by the outbound TCP processor.

31. (New) The hardware engine of Claim 29, wherein the outbound TCP processor requests the TCP Table Manager to read a network control block for a TCP connection from a host system memory and links the network control block to a hash value generated by a hash table maintained by the TCP Table manager.

32. (New) The hardware engine of Claim 29, wherein the TCP Table Manager includes a command processor that arbitrates between plural command sources and translates a received command to an output action(s) to other TCP Table Manager components.

33. (New) The hardware engine of Claim 32, wherein the TCP table manager's command processor co-ordinates inbound and outbound channel access to network control blocks.

34. (New) The hardware engine of Claim 29, wherein the TCP Table Manager provides a read / write register interface to allow simultaneous access to fields within the network control blocks.

35. (New) The hardware engine of Claim 34, wherein the TCP table manager's register interface provides a locking mechanism to allow sole access to a particular field within a network control block.

36. (New) The hardware engine of Claim 29, wherein the TCP table manager includes a timer list manager that maintains timer functions for all TCP connections that are handled by the hardware engine.

37. (New) The hardware engine of Claim 36, wherein the timer list manager maintains a persist timer, an idle timer, a delayed ACK timer and a retransmit timer for each TCP connection.

38. (New) The hardware engine of Claim 36, wherein the timer list manager scans a timer list and checks for timed events that have expired at any given time.

39. (New) The hardware engine of Claim 29, wherein the TCP table manager maintains an outbound request list to signal if a network control block is ready for transmit processing by the outbound TCP processor due to a timer event.

40. (New) The method of Claim 5, wherein the TCP Table Manager maintains timer functions for all TCP connections at any given time in the hardware engine and maintains a linked list of network control blocks for processing network data sent by the hardware engine and received by the hardware engine